Lösningar och kommentarer för tentamen 2025-03-13 i DA2004/5

Del A: flervalsfrågor

1. C

2. B

3. B

4. C

5. B

6. D

7. A

8. E

Del B: kodfrågor

9. Exempellösningar:

A.

```
def sum_vals(d):
s = 0
for k,v in d.items():
    if v % 2 == 0 and v > 0:
        s += v
return s
```

B.

```
def max_value(d):
    return max(d.values())
```

10. Exempellösningar:

A.

```
def read_numbers(h):
    l = []
    for line in h:
        try:
            x = int(line)
        except ValueError:
            continue
        l.append(x)
    return l
```

B.

```
def sum_file(fn):
    try:
        h = open(fn, 'r')
    except FileNotFoundError:
        return False
    return sum(read_numbers(h))
```

11. Exempellösningar.

En for och en while-loop:

```
def run_lengths(xs):
 result = []
 i = 0
 while i < len(xs): # Yttre loop: går igenom listan
     count = 1
     # Inre loop tittar framåt i listan
     for j in range(i + 1, len(xs)):
```

```
            if xs[j] == xs[i]:
                count += 1
            else:
                break
        result.append(count)
        i += count  # hoppa förbi hela gruppen
    return result
```

Två while-loopar:

```
   def run_lengths(xs):
    result = []
    i = 0
    while i < len(xs): # Yttre loop: går igenom listan
        count = 1
        j = i + 1
        # Inre loop räknar hur många gånger samma tal upprepas
        while j < len(xs) and xs[j] == xs[i]:
            count += 1
            j += 1
        result.append(count)
        i = j  # hoppa fram till nästa grupp
    return result
```

Med en for-loop:

```
def run_lengths(xs):
    rl = [] # initialisera
    curr_n = xs[0] # initialisera
    cnt = 1 # initialisera

    for i in range(1, len(xs)):
        if curr_n != xs[i]: # nytt tal
            rl.append(cnt)
            cnt = 1
            curr_n = xs[i]
        else:
            cnt += 1

    # behöver lägga till sista
    if curr_n == xs[-1]:
        rl.append(cnt)
    else:
        rl.append(1)
    return rl
```

12. Exempellösningar A och B:

13.

```
class GameCharacter:
    def __init__(self, name, role, level):
        self.name = name
        self.role = role
        self.level = level

    def __str__(self):
        return self.name + " (" + self.role + ") - level " + str(self.level)
        # Eller: return f"{self.name} ({self.role}) - level {self.level}"


class Party:
    def __init__(self, name, members):
        self.name = name
        self.members = members

        # Exempellösning för att kontrollera att alla roller är unika
        roles = [m.role for m in members]
        if len(roles) != len(set(roles)):
            raise ValueError("Each role must be unique within the party")
```